Expanded powertrain challenge problem

Prepared by Tunc Simsek, UC Berkeley, July 2002

Overview
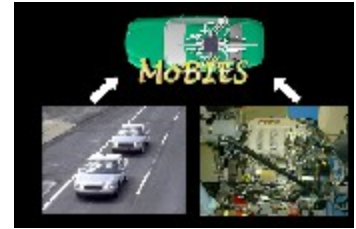
- **Previously looked at ETC**
  - Require simple 2nd order end-to-end performance
  - No requirements on software timing
- **Propose ETC/AFR control**
  - For the ETC part
    - **Same specifications**
  - For the AFR part
    - **Require maintaining .1 error of stoichiometric AFR**
    - **Stringent software timing requirements**
  - Model composition
    - **Obtain ETC/AFR models starting from separate ETC and AFR models**
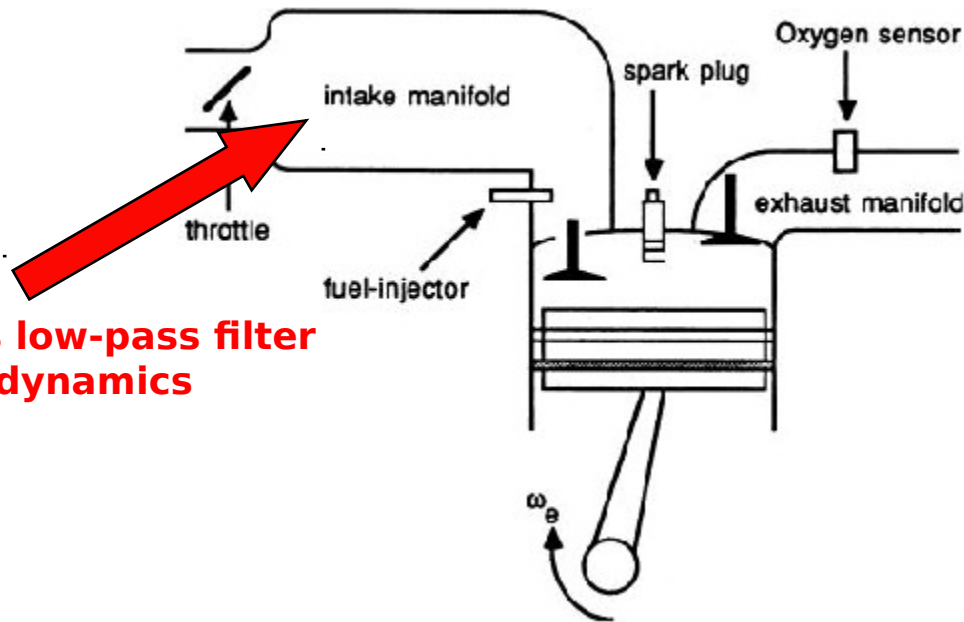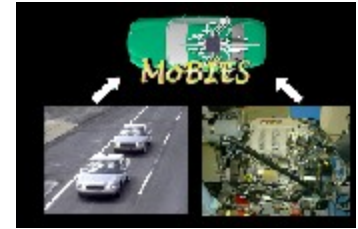
ETC requirement specification

Throttle angle (normalized) — overshoot, Rise-time, Settling-time, Time (10⁻¹s)
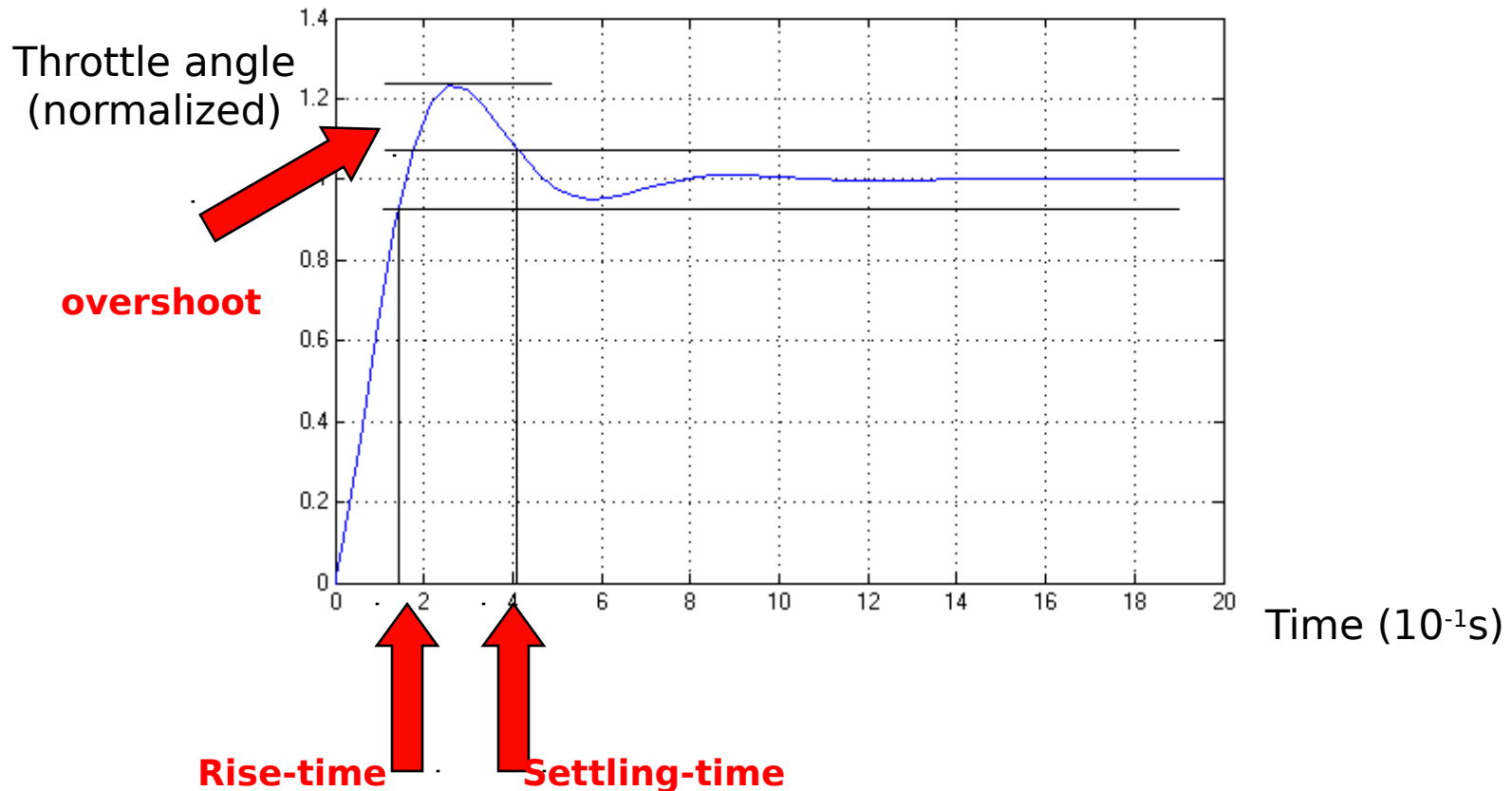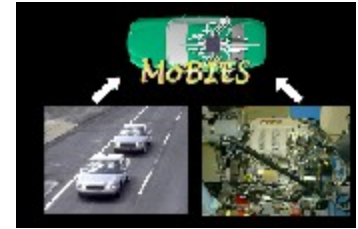
These are the end-to-end performance specifications
 – Significant w.r.t. human driver perception

ETC requirement specification

Reservoir acts as low-pass filter for airflow dynamics
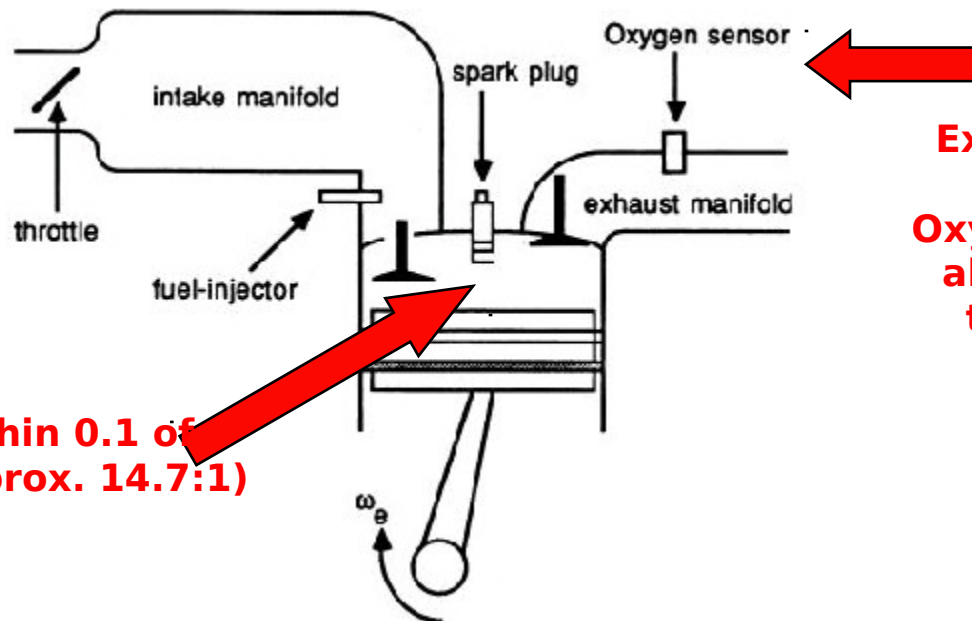
- **Throttle body has almost linear dynamics**
- **Air-flow dynamics at intake-port insensitive to actual throttle timing**
  - Can use straightforward uniformly sampled discrete-time controller
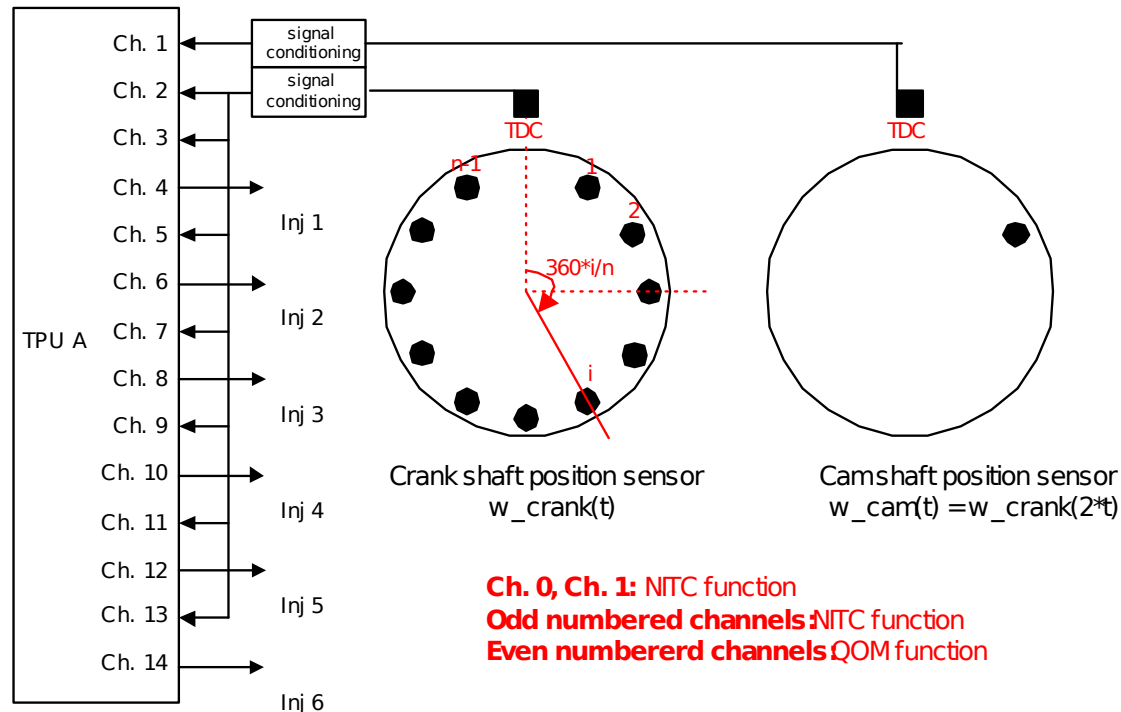
AFR requirement specification

Exact stoichiometric ratio unknown. Oxygen sensor gives 1 above and –1 below the desired ratio.

Require AFR to be within 0.1 of stoichiometric ratio (approx. 14.7:1)

- **Combustion dynamics nowhere linear**
- **Mixture forming dynamics very sensitive to AFR software**
  - relative to the valve positition (open or closed),
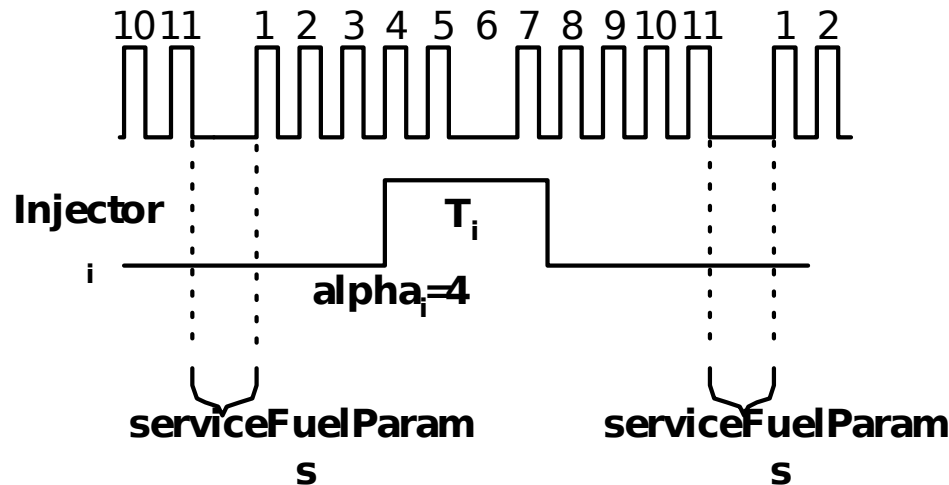 - and relative to the amount of fuel sprayed

Fuel-injector actuation

Crank shaft position sensor
w_crank(t)

Camshaft position sensor
w_cam(t) = w_crank(2*t)

Ch. 0, Ch. 1: NITC function
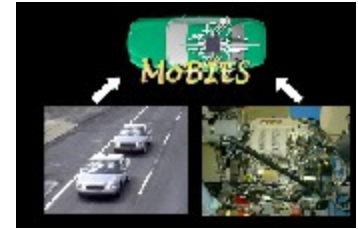**Odd numbered channels:** NITC function
Even numbererd channels: QOM function
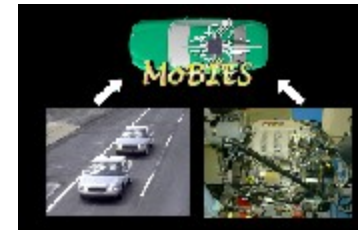
- **Actuator timing relative to crank/cam shaft**
- **Need high timing resolution**
- **Cannot use the CPU for this purpose**
 - Time Processor Unit (TPU) co-processor is used to control the timing

Fuel-injector actuation

10 11   1 2 3 4 5 6 7 8 9 10 11   1 2

Injector

T_i

i

$alpha_i = 4$

serviceFuelParams

serviceFuelParams

- **Example with 5 (+1 missing) crank teeth**
- **Fuel injectors on/off actuated**
- **Interface to software through an interrupt service routine (serviceFuelParams)**
  - The ISR updates $alpha_i$ and T_i for i^{th} injector
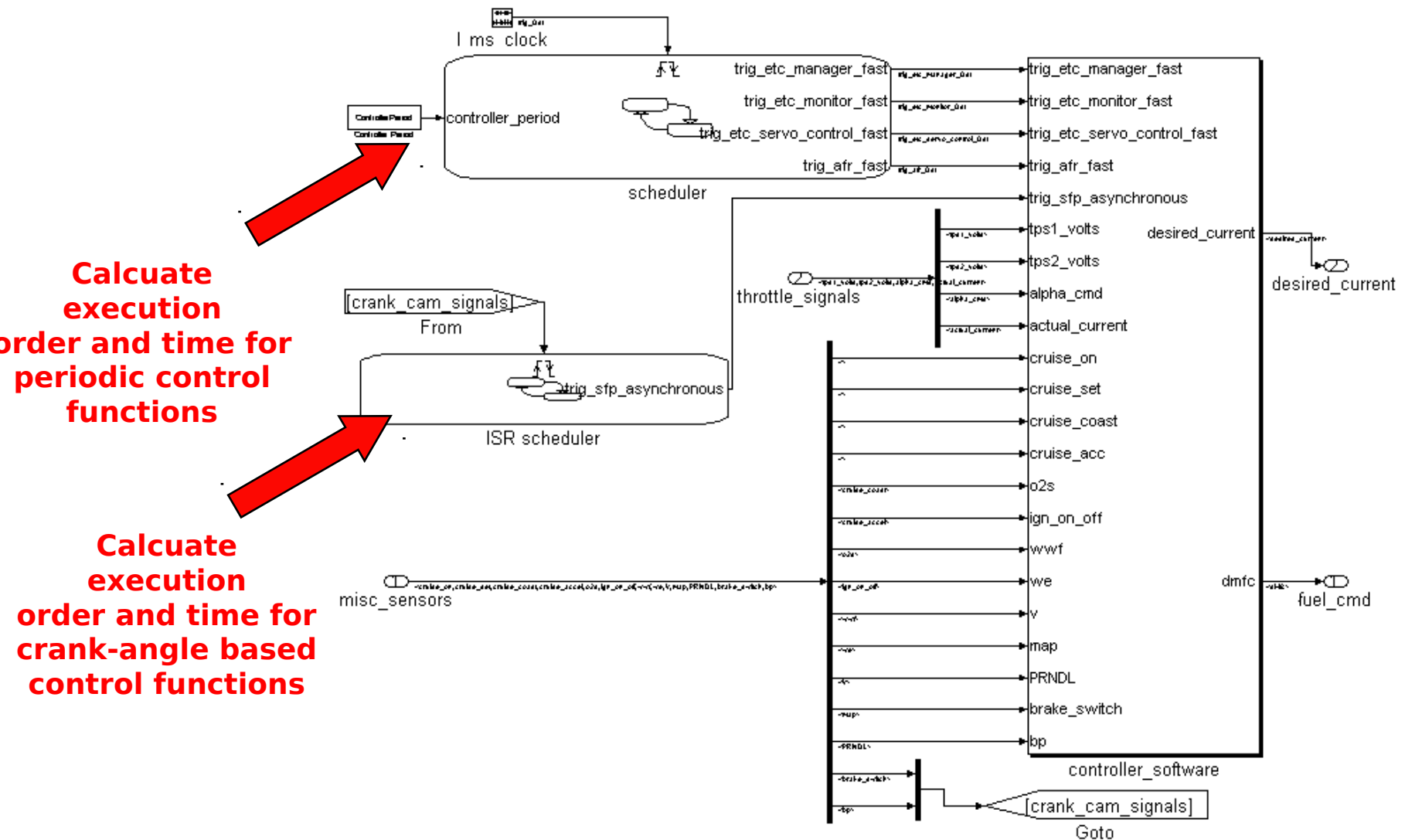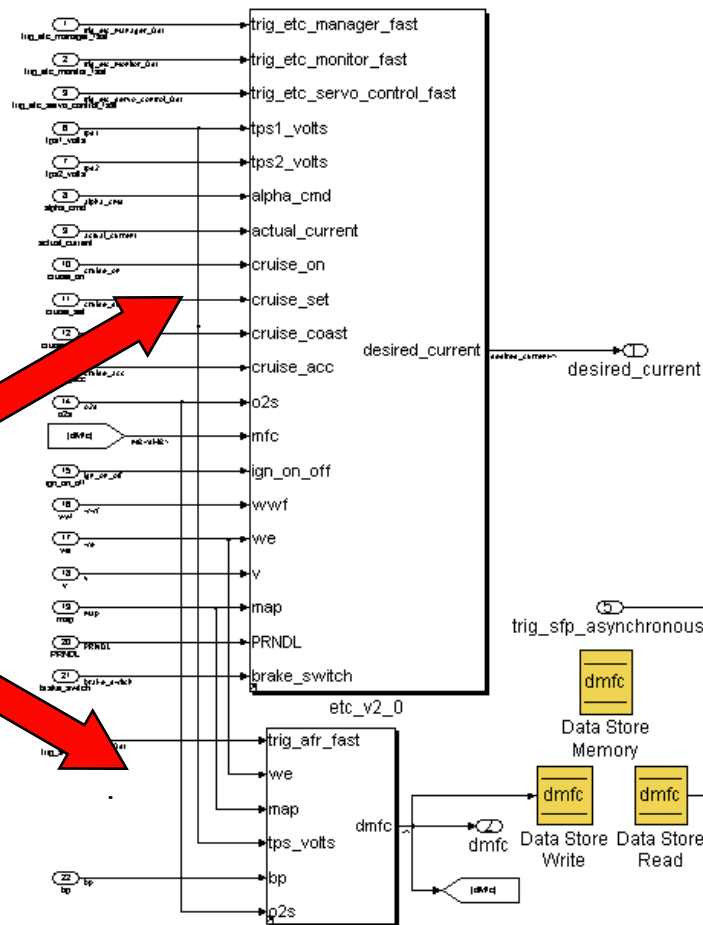 - behavior of TPU is otherwise autonomous

Status
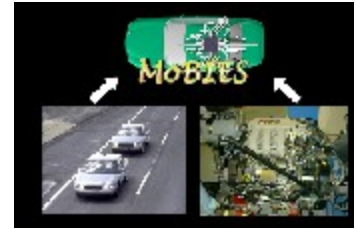
- **Current baseline:**
  - ETC/AFR models in Simulink/Stateflow with serviceFuelParams given as S-function
  - Methodology for mapping model components to OSEK objects (tasks, resources, ISRs)
  - TPU micro code for crank/cam sensing and fuel-injector actuation
- **Next step:**
  - PIL simulator

Calcuate execution order and time for periodic control functions

Calcuate execution order and time for crank-angle based control functions

Functional controller software

ETC controller
(periodic)

AFR controller
(periodic)

serviceFuelParams

Data objects

Explicit data stores mapped to shared OSEK resources
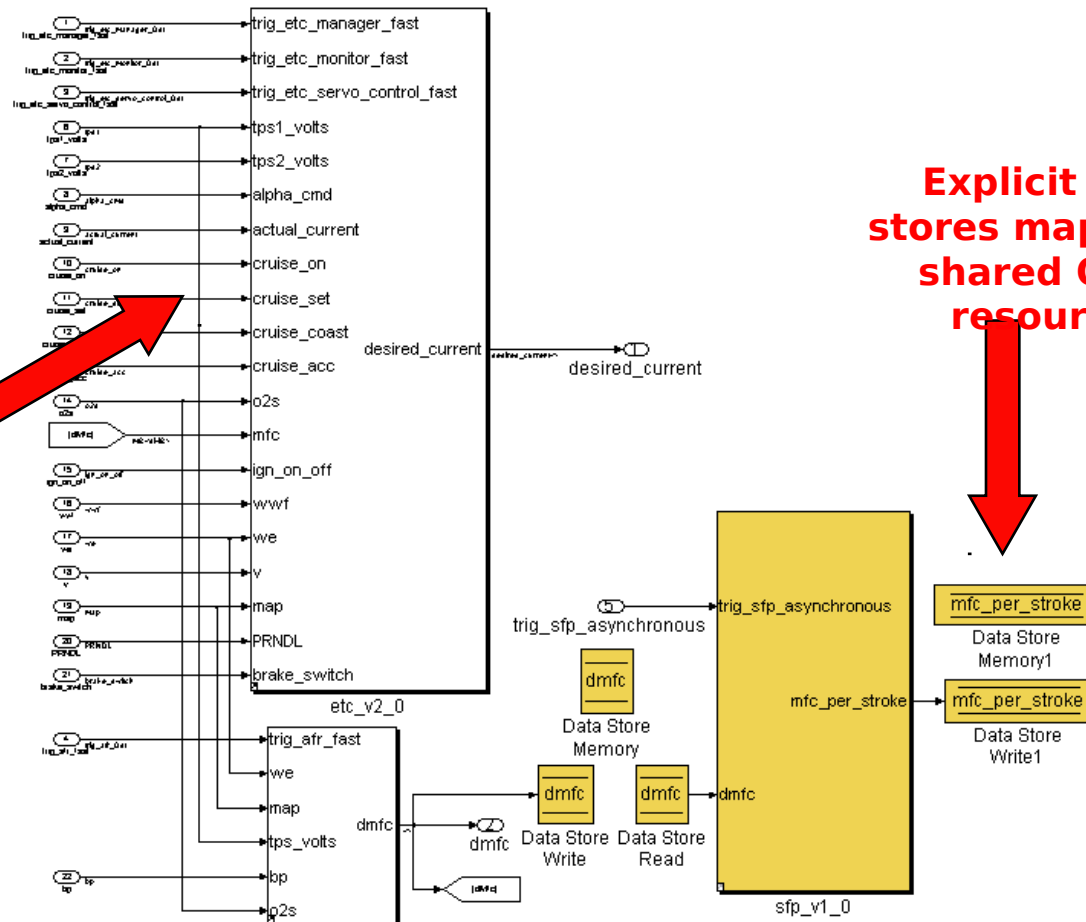
All wires are data stores (mapped to global variables)
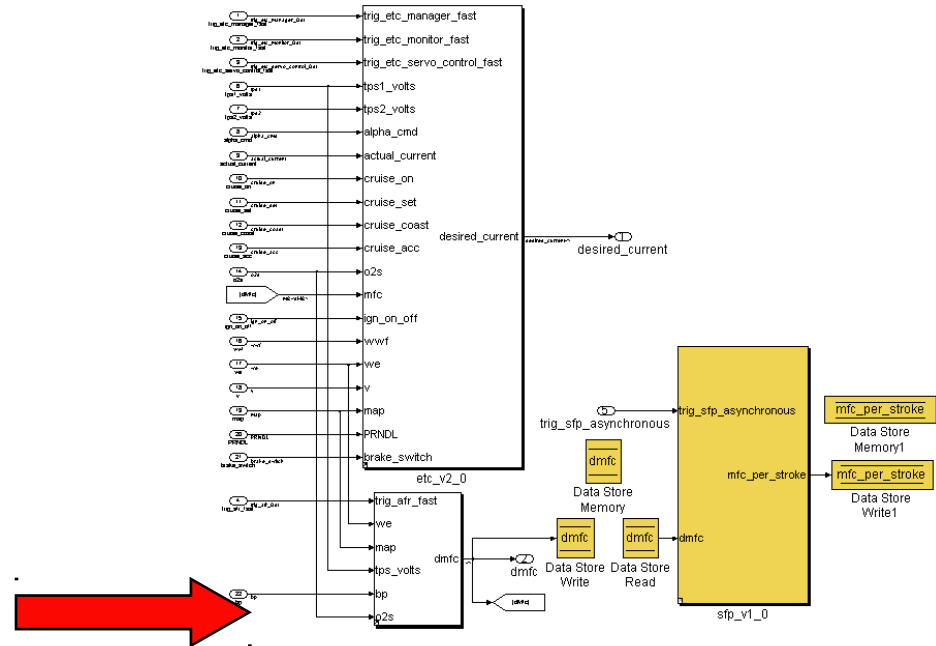
Tasks

```
TASK(afr_v1_0) {
    while(1) {
        WaitEvent(trigger_afr_fast);
        ClearEvent(trigger_afr_fast);
        AcquireInputs_afr_v1_0();
        GetResource(dmfc_lock);
        OutputUpdate_afr_v1_0();
        ReleaseResource(dmfc_lock);
        SetEvent(afr_v1_0_states,
stateUpdate);
    }
}

TASK(afr_v1_0_states) {
    while(1) {
        WaitEvent(stateUpdate);
        ClearEvent(stateUpdate);
        StateUpdate_afr_v1_0();
    }
}
```
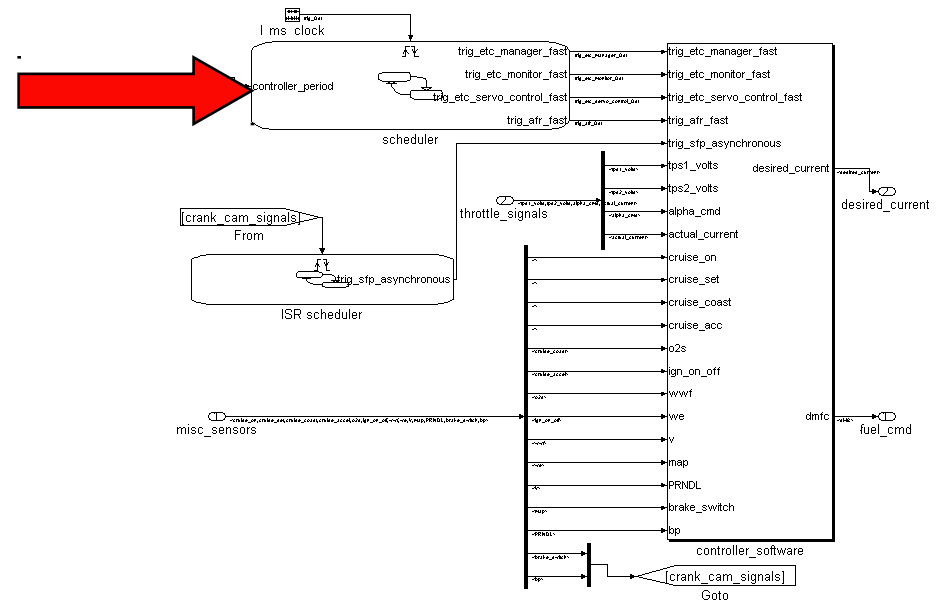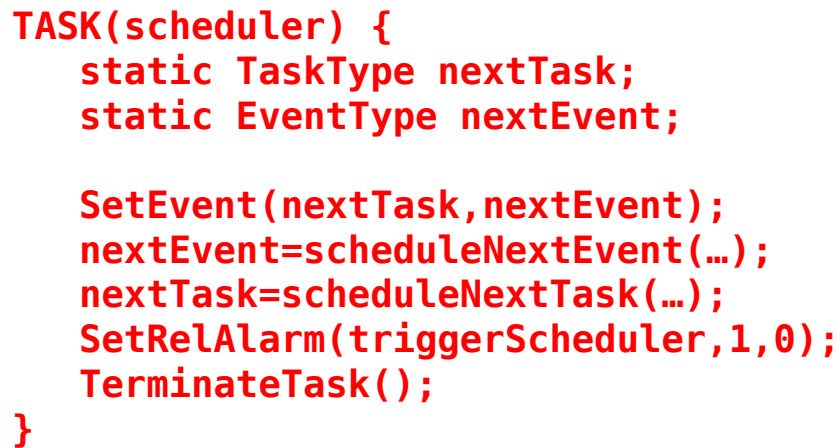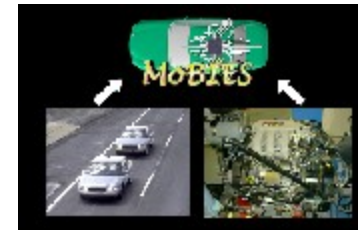
Scheduler (periodic)

```
TASK(scheduler) {
    static TaskType nextTask;
    static EventType nextEvent;

    SetEvent(nextTask,nextEvent);
    nextEvent=scheduleNextEvent(…);
    nextTask=scheduleNextTask(…);
    SetRelAlarm(triggerScheduler,1,0);
    TerminateTask();
}
```

Scheduler (aperiodic)

```
ISR(ISRscheduler) {
    SetEvent(sfp_v1_0, trig_sfp_asynchronous);
}


TASK(sfp_v1_0) {
    while(1) {
        WaitEvent(trigger_sfp_asynchronous);
        ClearEvent(trigger_sfp_asyncronous);
        GetResource(dmfc_lock);
        AcquireInputs_sfp_v1_0();
        ReleaseResource(dmfc_lock);
        OutputUpdate_sfp_v1_0();
        SetEvent(sfp_v1_0_states, stateUpdate);
    }
}


TASK(sfp_v1_0_states) {
    while(1) {
        WaitEvent(stateUpdate);
        ClearEvent(stateUpdate);
        StateUpdate_sfp_v1_0();
    }
}
```
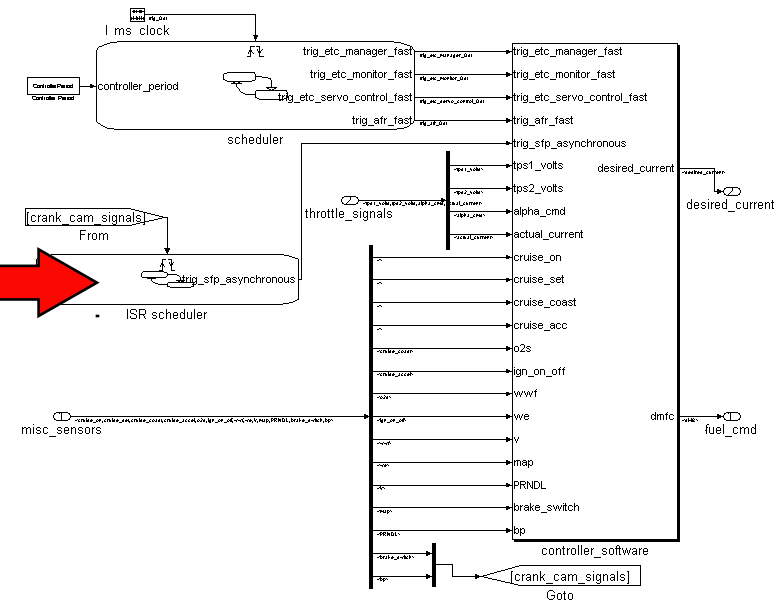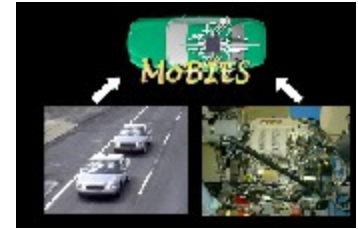
Summary of baseline tools

- **Targetlink, Embedded Coder**
  - functional code
- **Rapidhawk, New Eagle Systems**
  - Simulink/stateflow to OSEK solution
  - A Simulink toolkit with icons representing OSEK objects
  - Want to use Rapidhawk to produce code for
    - **Data acquisition (A/D, TPU, …)**
    - **PIL drivers**
- **Still need baseline analysis tools**